Appl. Math. Lett. Vol. 4, No. 6, pp. 27–31, 1991
Printed in Great Britain.

0893-9659/91 $3.00 + 0.00

Parallel Algebraic Reductions among Numerical Problems

[1]B. CODENOTTI, [2]M. LEONCINI, AND [1]G. RESTA

[1]Istituto di Elaborazione dell'Informazione, Consiglio Nazionale delle Ricerche
[2]Dipartimento di Informatica, Università di Pisa

(Received March 1991)

Abstract. In this note we consider, for a number of linear algebra problems, an environment allowing approximate computations. Within this framework we show that the relative complexity of these problems should be studied according to a strict notion of reducibility, which corresponds to the well-known many-one reducibility of combinatorial complexity.

1. INTRODUCTION

The concept of *resource bounded reduction* has played a major role in deepening the understanding on the complexity of computational problems, in both combinatorial and algebraic settings. Generally speaking, to say that a problem A is *reducible* to a problem B amounts to saying that an algorithm R can be found which solves A given (e.g., as a subroutine to R) an algorithm R_B for B. When the computational cost of R is not higher than that of R_B, we can immediately draw the conclusion that A *is not harder than* B. Thus reducibility is one of the most important theoretical tools suitable to classify problems basing upon their demand of computational resources (e.g., time or space).

We point out that, in computational complexity, problems have almost always been classified according to the cost of the available "exact" solution methods. However, many important problems are not feasibly solvable in an exact way, and in fact there exists a huge body of literature on approximate solution methods, again for both combinatorial and algebraic problems. It is then pretty natural to ask whether and how the concept of approximation can affect the complexity hierarchies given by the known reductions. Motivated by this question, we present here some preliminary results on parallel reducibility within approximation settings which apply to a number of linear algebra problems.

Let F be a field. We are concerned with the following problems defined over F:

- $DET_F(n)$: computing the determinant of an $n \times n$ matrix;
- $CHARPOLY_F(n)$: computing the characteristic polynomial of an $n \times n$ matrix;
- $ADJOINT_F(n)$: computing the adjoint matrix of an $n \times n$ matrix;
- $INVERSE_F(n)$: computing the inverse of a nonsingular $n \times n$ matrix;
- $NONSINGEQ_F(n)$: solving a nonsingular system of n linear equations in n unknowns;
- $POWERS_F(n)$: computing the powers 2 through n of an $n \times n$ matrix;
- $INTEPROD_F(n)$: computing the product of n $n \times n$ matrices†.

In [1] Csanky shows that the above problems can be solved, when F is a field of characteristic 0, in $O(\log^2 n)$ depth and polynomial (in n) size, i.e. that they belong to the complexity class NC^2_F. (For a definition of NC_F and NC^k_F see [2].) Csanky's result has been later improved by various authors, so that it is now possible to achieve the same depth-size bound over more general algebraic structures, such as arbitrary fields or even, for what concerns $DET(n)$ and $CHARPOLY(n)$, commutative rings with unity [3–5].

We have been unable to communicate with the author with respect to galley proof corrections. Hence, this work is published without the benefit of such corrections. (Ed.)

†For the sake of brevity, in the following we write $DET(n)$ for $DET_F(n)$ (and analogously for the other problems).

2. REDUCTIONS IN AN EXACT ENVIRONMENT

In [1] Csanky also presents $O(\log n)$ time reductions among some of the above defined problems. In this early work, the model of computation adopted was somewhat informal. Here we consider the more rigorous model of *arithmetic circuits* [2,6], and give two informal definitions of resource bounded reduction, which correspond to the well known notions of polynomial (or Turing) reduction and polynomial transformation (or many-one reduction).

DEFINITION 1. (NC_F^1 REDUCTION). *A problem $P(n)$ is NC_F^1 reducible to $R(m)$ if there is an NC_F^1 circuit which solves $P(n)$ using oracle nodes for $R(i), i \in \{1, \ldots, m\}, m = n^{O(1)}$. An oracle node with k inputs counts $O(\log k)$ depth.* ∎

If $P(n)$ is NC_F^1 reducible to $R(m)$ we write $P(n) \leq_T^p R(m)$.

DEFINITION 2. (NC_F^1 TRANSFORMATION). *A problem $P(n)$ NC_F^1 transforms to $R(m)$ if there exists an NC_F^1 circuit α which transforms an instance $\mathbf{x}$ of $P(n)$ to an instance $\alpha(\mathbf{x})$ of $R(m)$, and such that any solution to $\alpha(\mathbf{x})$ can be interpreted as a solution to $\mathbf{x}$.* ∎

If $P(n)$ NC_F^1 transforms to $R(m)$, we write $P(n) \leq_m^p R(m)$.

Since the definition of NC_F^1 reduction is more general, if $P(n) \leq_m^p R(m)$ holds, then also $P(n) \leq_T^p R(m)$ holds. The converse is not necessarily true, in general. We later show that the notion suitable to the case of approximating algorithms is that of NC_F^1 transformation.

Two problems $P(n)$ and $R(m)$ are said to be $\leq_m^p$ ($\leq_T^p$) *equivalent* if both $P(n) \leq_m^p R(m)$ and $R(n) \leq_m^p P(k)$ ($P(n) \leq_m^p R(m)$ and $R(n) \leq_m^p P(k)$) hold. In this case we write $P \equiv_m^p R$ ($P \equiv_T^p R$). Note that, within the NC_F theory, it is sufficient that $m, k = n^{O(1)}$. The known reductions allow to conclude that all the problems introduced in the previous section are $\leq_T^p$ equivalent. On the other hand, the reductions that we can show to be of the $\leq_m^p$ type are listed below.

(1) $INVERSE(n) \leq_m^p NONSINGEQ(n^2)$ and $NONSINGEQ(n) \leq_m^p INVERSE(n)$;
(2) $DET(n) \leq_m^p ADJOINT(n+1)$;
(3) $DET(n) \leq_m^p CHARPOLY(n)$;
(4) $POWERS(n) \leq_m^p ITEPROD(n)$ and $ITEPROD(2n-1) \leq_m^p POWERS(n^2)$.

3. REDUCTIONS IN PRESENCE OF APPROXIMATION ERRORS

As previously stated, in past studies on the complexity of algebraic problems, the general notion of NC_F^1 reduction was adopted (see [2]). Below we show that NC_F^1 transformation is indeed necessary when dealing with approximation. We begin by defining (in an informal way) the equivalent of NC_F when approximation is allowed.

DEFINITION 3. *An arithmetic circuit family $\alpha = \{\alpha_n\}_{n \in N}$ over the field F is an ϵ-Approximating Arithmetic Circuit (or simply α_ϵ-circuit) family for a problem P if and only if for all possible inputs of size n the relative error (according to a well-defined measure) by which the output of α_n is affected with respect to an exact solution is bounded by ϵ.* ∎

According to the NC_F theory, we define NC_ϵ^k, for all $\epsilon > 0$ and integers $k \geq 1$, as the class of problems solvable by α_ϵ-circuit families of $O(\log^k \textit{input size})$ depth and $(\textit{input size})^{O(1)}$ size. The following proposition can be easily proved, provided that the relative error bound ϵ is independent of the problem size. This is the main reason for using NC_F^1 transformations.

PROPOSITION 4. *Let $\epsilon > 0$ and let $k \geq 1$ be an integer. Then $R \in NC_\epsilon^k$ and $P \leq_m^p R$ imply $P \in NC_\epsilon^k$.* ∎

Proposition 4 states that the $\leq_m^p$ relation is transitive with respect to the error limitations. We now show that the reductions (1), (2), and (4) are indeed of $\leq_m^p$ type (the proof of (3) is trivial).

PROPOSITION 5.

a) $INVERSE(n) \leq_m^p NONSINGEQ(n^2)$, *and*
b) $NONSINGEQ(n) \leq_m^p INVERSE(n)$.

PROOF:

a) To invert the matrix A, form the $n^2 \times n^2$ block diagonal linear system

$$\mathrm{diag}(A, \ldots, A) = (\mathbf{e}_1^T | \ldots | \mathbf{e}_n^T)^T.$$

The n^2 elements of the vector solution are the elements of the matrix A^{-1} (in column major order). Moreover, with respect to the most commonly used matrix norms, the condition number of the matrix $\mathrm{diag}(A, \ldots, A)$ is equal to that of A.

b) To reduce $NONSINGEQ(n)$ to $INVERSE(n)$, consider the $n \times n$ linear system $A\mathbf{x} = \mathbf{b}$. Assume, without loss of generality, that $|b_1| = \max\{|b_i| : i = 1, \ldots, n\} = 1$ (the condition $|b_1| = 1$ can be easily satisfied by scaling). With a step of Gaussian elimination, in constant time and without loss of precision, obtain the equivalent system $A'\mathbf{x} = \mathbf{e}_1$. Then the solution of the system coincides with the first column of A'^{-1}. The condition number of A', with respect to the infinity norm, is bounded by the quantity $\mu(A)\mu(E)$, where $\mu(E)$ is the condition number of the matrix

$$E = \begin{pmatrix} 1 & & & \\ -b_2 & 1 & & \\ \vdots & & \ddots & \\ -b_n & & & 1 \end{pmatrix},$$

which describes the step of Gaussian elimination. It is easy to see that $||E||_\infty < 2$ and $||E^{-1}||_\infty < 2$, where $||\cdot||_\infty$ denotes the infinity matrix norm. ∎

The reason for studying the effect of $\leq_m^P$ reductions on the condition number of matrices is that the behaviour of approximating (e.g. iterative) algorithms depends on such a measure, as well as on the input size.

PROPOSITION 6. $DET(n) \leq_m^P ADJOINT(n+1)$.

PROOF: Given an order n matrix A, we consider the $(n+1) \times (n+1)$ matrix

$$B = \begin{pmatrix} A & 0 \\ 0^T & 1 \end{pmatrix}.$$

It is then easy to see that

$$\mathrm{adj}(B) = \begin{pmatrix} \det(A)A^{-1} & 0 \\ 0^T & \det(A) \end{pmatrix}. \tag{1}$$

Strictly speaking, this is not an NC_F^1 transformation, since the result is one element of the adjoint and not the adjoint itself. The determinant can be obtained from the adjoint by a single use of projection†. However, from the standpoint of the error analysis, it is necessary that no arithmetic computation be performed on the result of the transformed problem, and clearly projections meet this requirement. ∎

PROPOSITION 7.

a) $POWERS(n) \leq_m^P INTEPROD(n)$, *and*
b) $ITEPROD(2n-1) \leq_m^P POWERS(n^2)$. ∎

†We recall that a projection $p_i : F^k \to F$ is defined as follows: $p_i(x_1, \ldots, x_k) = x_i$, for any $k \geq 1$ and $1 \leq i \leq k$.

SKETCH OF THE PROOF: The proof of *a*) is straightforward. To prove *b*), consider the following $n^2 \times n^2$ matrix

$$M = \begin{pmatrix} B & C_1 & O & \dots & O \\ A_1 & O & \ddots & \ddots & \vdots \\ O & \ddots & \ddots & C_{n-2} & O \\ \vdots & \ddots & A_{n-2} & O & C_{n-1} \\ O & \dots & O & A_{n-1} & O \end{pmatrix},$$

where $A_i, C_i, i = 1, \dots, n-1$, and B are $n \times n$ matrices. It can be shown that the $n \times n$ matrix in the bottom right corner of M^{2n-1} is the product $A_{n-1}A_{n-2} \cdots A_1 B C_1 C_2 \cdots C_{n-1}$. ∎

We do not know of any NC_F^1 transformation which relates (in either directions) from one side matrix inversion and linear system solution, from the other the computation of determinant, adjoint and characteristic polynomial of a matrix. In particular, it can be shown that no NC_F^1 transformation $INVERSE(n) \leq_m^p ADJOINT(n)$ exists. In fact, given a matrix A, we should determine a matrix B such that $\text{adj}(B) = A^{-1}$. But this implies $B = \sqrt[n-1]{1/\det(A)}A$, and the quantity $\sqrt[n-1]{1/\det(A)}$ is not NC_F^1 computable.

Problems that are equivalent under NC_F^1 transformations appear strongly related even in an approximation environment. In some sense, approximation makes it apparent the different nature of the problems under investigation. We do know iterative (approximate) processes for solving $NONSINGEQ(n)$ and, by Propositions 4 and 5, $INVERSE(n)$ in $NC_{\mathbf{R}}^1$ (**R** denotes the real field), but we do not know how to take advantage of approximation for solving problems with a sort of combinatorial flavour, such as DET (see [7]).

By taking the more general notion of NC_F^1 reduction into account, other relations can be considered which can be of some help also in an approximation environment. We know that Proposition 4 does not hold for $\leq_T^p$ reductions. In fact, according to Definition 1, an NC_F^1 reduction allows both a polynomial number of oracle gates and a polynomial number of operations to be performed on the oracle outputs. From the viewpoint of approximation (i.e. in order to succeed in producing a result which is affected by a relative error bounded by a given constant ϵ), this fact may require the error bound at the oracle nodes to be of the form $\frac{\epsilon}{n^{O(1)}}$. Consider, for instance, the well-known reduction from $DET(n)$ to $LU(n)$. This consists of multiplying the diagonal elements of the U factor produced by the oracle for $LU(n)$. However, this can decrease the precision of the result by a factor $O(n)$ (n = size of the input matrix).

Under NC_F^1 reductions, we are able to relate $INVERSE(n)$ and $ADJ(n+1)$ by means of the formula (1). The i,j-th element of A^{-1} can be easily obtained by projecting the i,j-th and the $n+1, n+1$-th element of $\text{adj}(B)$ and then computing the quotient, $i, j = 1, \dots, n$. The error bound on the result is only "slightly" increased with respect to the error introduced by the oracle for $ADJ(n+1)$. In fact, the following proposition can be proved.

PROPOSITION 8. *Let A be a nonsingular $n \times n$ matrix, and let $\mathcal{A} = \text{adj}(A)$ and $d = \det(A)$. Let $\mathcal{A}'$ and d' be approximations of $\mathcal{A}$ and d that satisfy*

$$\frac{||\mathcal{A} - \mathcal{A}'||}{||\mathcal{A}||} < \epsilon_1 \quad \text{and} \quad \frac{|d - d'|}{|d|} < \epsilon_2,$$

where the relative errors ϵ_1 and ϵ_2 are supposed to be independent from n. Then the approximation $\frac{\mathcal{A}'}{d'}$ of $\frac{\mathcal{A}}{d} = A^{-1}$ is affected by a relative error $\epsilon_3 = O(\epsilon_1 + \epsilon_2)$. ∎

REFERENCES

1. L. Csanky, Fast parallel matrix inversion algorithms, *SIAM J. Comput.* 5, 618–623 (1976).
2. J. von zur Gathen, Parallel arithmetic computations: A survey, *Lecture Notes in Computer Science* Vol. 233, Springer-Verlag, Berlin, New York, 93–122, (1986).

3. S.J. Berkowitz, On computing the determinant in small parallel time using a small number of processors, *Inform. Process. Lett.* **18**, 147–150 (1984).
4. A. Borodin, J. von zur Gathen, and J. Hopcroft, Fast parallel matrix and GCD computations, *Inform. and Control* **52**, 241–256 (1982).
5. K. Mulmuley, A fast parallel algorithm to compute the rank of a matrix over an arbitrary field, *Proc. 18th Annual ACM Symposium on Theory of Computing*, 338–339 (1986).
6. A. Borodin, On relating time and space to size and depth, *SIAM J. Comput.* **6**, 733–744 (1977).
7. B. Codenotti, M. Leoncini, and G. Resta, Parallel complexity and approximation, submitted for publication.

[1]Istituto di Elaborazione dell'Informazione, Consiglio Nazionale delle Ricerche, Pisa, Italy
[2]Dipartimento di Informatica, Università di Pisa, Pisa, Italy